```
                                  H  5

FFFFFFFFFFFFFFF        111               111        XXX         XXX
FFFFFFFFFFFFFFF        111               111        XXX         XXX
FFFFFFFFFFFFFFF        111               111        XXX         XXX
FFF                 111111            111111        XXX         XXX
FFF                 111111            111111        XXX         XXX
FFF                 111111            111111        XXX         XXX
FFF                    111               111           XXX   XXX
FFF                    111               111           XXX   XXX
FFF                    111               111           XXX   XXX
FFFFFFFFFFFF           111               111              XXX
FFFFFFFFFFFF           111               111              XXX
FFFFFFFFFFFF           111               111              XXX
FFF                    111               111           XXX   XXX
FFF                    111               111           XXX   XXX
FFF                    111               111           XXX   XXX
FFF                    111               111        XXX         XXX
FFF                    111               111        XXX         XXX
FFF                    111               111        XXX         XXX
FFF                    111               111        XXX         XXX
FFF                 111111111         111111111     XXX         XXX
FFF                 111111111         111111111     XXX         XXX
FFF                 111111111         111111111     XXX         XXX
```

```
_$25

Symb
----
IOC$
IO_C
IO_C
IO_D
IO_F
IO_S
KILL

KILL
KILL
LB_E
LB_D
LB_F
LB_H
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CO
L_CO
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP
MARK
MARK
MARK
MARK
MARK
```

```
**FILE**ID**ACLCNTRL




  AAAAAA      CCCCCCCC  LL            CCCCCCCC  NN      NN  TTTTTTTTTT  RRRRRRRR    LL
  AAAAAA      CCCCCCCC  LL            CCCCCCCC  NN      NN  TTTTTTTTTT  RRRRRRRR    LL
AA      AA  CC          LL          CC          NN      NN      TT      RR      RR  LL
AA      AA  CC          LL          CC          NN      NN      TT      RR      RR  LL
AA      AA  CC          LL          CC          NNNN    NN      TT      RR      RR  LL
AA      AA  CC          LL          CC          NNNN    NN      TT      RR      RR  LL
AA      AA  CC          LL          CC          NN  NN  NN      TT      RRRRRRRR    LL
AA      AA  CC          LL          CC          NN  NN  NN      TT      RRRRRRRR    LL
AAAAAAAAAA  CC          LL          CC          NN    NNNN      TT      RR  RR      LL
AAAAAAAAAA  CC          LL          CC          NN    NNNN      TT      RR  RR      LL
AA      AA  CC          LL          CC          NN      NN      TT      RR    RR    LL              ....
AA      AA  CC          LL          CC          NN      NN      TT      RR    RR    LL              ....
AA      AA    CCCCCCCC  LLLLLLLLLL    CCCCCCCC  NN      NN      TT      RR      RR  LLLLLLLLLL      ....
AA      AA    CCCCCCCC  LLLLLLLLLL    CCCCCCCC  NN      NN      TT      RR      RR  LLLLLLLLLL      ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;   1        0001  0 MODULE ACLCNTRL (
;   2        0002  0                 LANGUAGE (BLISS32),
;   3        0003  0                 IDENT = 'V04-000'
;   4        0004  0                 ) =
;   5        0005  1 BEGIN
;   6        0006  1
;   7        0007  1 !
;   8        0008  1 !****************************************************************************
;   9        0009  1 !*                                                                          *
;  10        0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;  11        0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
;  12        0012  1 !*  ALL RIGHTS RESERVED.                                                    *
;  13        0013  1 !*                                                                          *
;  14        0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;  15        0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
;  16        0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;  17        0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;  18        0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;  19        0019  1 !*  TRANSFERRED.                                                            *
;  20        0020  1 !*                                                                          *
;  21        0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;  22        0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;  23        0023  1 !*  CORPORATION.                                                            *
;  24        0024  1 !*                                                                          *
;  25        0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;  26        0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;  27        0027  1 !*                                                                          *
;  28        0028  1 !*                                                                          *
;  29        0029  1 !****************************************************************************
;  30        0030  1
;  31        0031  1 !++
;  32        0032  1 !
;  33        0033  1 ! FACILITY:      F11ACP Structure Level 2
;  34        0034  1 !
;  35        0035  1 ! ABSTRACT:
;  36        0036  1 !
;  37        0037  1 !       This module contains the routines to manipulate the Access
;  38        0038  1 !       Control Lists and Access Control Entries.  This includes all
;  39        0039  1 !       of the functionality necessary for the ACL editing.
;  40        0040  1 !
;  41        0041  1 ! ENVIRONMENT:
;  42        0042  1 !
;  43        0043  1 !       STARLET operating system, including privileged system services
;  44        0044  1 !       and internal exec routines. This routine must be executed
;  45        0045  1 !       in kernel mode.
;  46        0046  1 !
;  47        0047  1 !--
;  48        0048  1
;  49        0049  1 !
;  50        0050  1 ! AUTHOR:        L. Mark Pilant,          CREATION DATE:  30-Jun-1982  9:30
;  51        0051  1 !
;  52        0052  1 ! MODIFIED BY:
;  53        0053  1 !
;  54        0054  1 !       V03-032 LMP0306         L. Mark Pilant,         29-Aug-1984  9:24
;  55        0055  1 !               Disallow ACLs on the index file.
;  56        0056  1 !
;  57        0057  1 !       V03-031 ACG0446         Andrew C. Goldstein,    21-Aug-1984  22:01
;  58          0058   1 !                Clean out ACL residue possibly left by other file system
;  59          0059   1 !                versions
;  60          0060   1 !
;  61          0061   1 !        V03-030 LMP0284         L. Mark Pilant,         25-Jul-1984  14:53
;  62          0062   1 !                Move ACL_INIT_QUEUE to ACLSUBR.  Also, only initialize the
;  63          0063   1 !                queue if it is being modified.
;  64          0064   1 !
;  65          0065   1 !        V03-029 LMP0275         L. Mark Pilant,         19-Jul-1984  13:09
;  66          0066   1 !                If the ACL queue is uninitialized in the ORB (in the FCB),
;  67          0067   1 !                it is necessary to initialize it before any ACL operations
;  68          0068   1 !                can take place.
;  69          0069   1 !
;  70          0070   1 !        V03-028 ACG0437         Andrew C. Goldstein,    13-Jul-1984  16:43
;  71          0071   1 !                Remove ACL_COPYACL call to ACL_BUILDACL (to a higher level)
;  72          0072   1 !
;  73          0073   1 !        V03-027 LMP0243         L. Mark Pilant,         27-Apr-1984  14:23
;  74          0074   1 !                Allow READACE to bypass BADACL.  Also, don't abort ACL attr
;  75          0075   1 !                processing if an error occurs;  only disallow modifications.
;  76          0076   1 !
;  77          0077   1 !        V03-026 ACG0415         Andrew C. Goldstein,     4-Apr-1984  10:07
;  78          0078   1 !                Break out common ACL subroutines; fix space allocation of
;  79          0079   1 !                ACL in last header used; fix space left for map pointer
;  80          0080   1 !                left in first header. Check for bad ACL on most operations.
;  81          0081   1 !
;  82          0082   1 !        V03-025 LMP0216         L. Mark Pilant,         22-Mar-1984  12:06
;  83          0083   1 !                Correct a bug introduced in LMP0174 that caused the DEFAULT
;  84          0084   1 !                bit in the ACE flags to be cleared when propagating the ACL
;  85          0085   1 !                for a directory file. Also make rest of global storage based.
;  86          0086   1 !                Rename locals to avoid conflicts with globals.
;  87          0087   1 !
;  88          0088   1 !        V03-024 LMP0210         L. Mark Pilant,         10-Mar-1984  10:36
;  89          0089   1 !                Add support for the ACL status field.
;  90          0090   1 !
;  91          0091   1 !        V03-023 CDS0003         Christian D. Saether     7-Mar-1984
;  92          0092   1 !                Don't mark FILE_HEADER dirty in ACL_BUILDACL if
;  93          0093   1 !                the checksum was already OK.
;  94          0094   1 !
;  95          0095   1 !        V03-022 CDS0002         Christian D. Saether    27-Dec-1983
;  96          0096   1 !                Use BIND_COMMON macro.  Adjust routine and external
;  97          0097   1 !                declarations.
;  98          0098   1 !
;  99          0099   1 !        V03-021 CDS0001         Christian D. Saether    12-Dec-1983
; 100          0100   1 !                Move all OWN storage to COMMON module to reduce xqp
; 101          0101   1 !                to single image section.
; 102          0102   1 !
; 103          0103   1 !        V03-020 LMP0174         L. Mark Pilant,          1-Dec-1983  14:25
; 104          0104   1 !                Change the name of the routine to propagate default ACEs.
; 105          0105   1 !                Also, make it a little more general purpose in terms of
; 106          0106   1 !                the types of propagation it is able to do.
; 107          0107   1 !
; 108          0108   1 !        V03-019 LMP0159         L. Mark Pilant,         30-Sep-1983  9:29
; 109          0109   1 !                Use the correct base address when clearing out the ACL
; 110          0110   1 !                portion of the file header during a build operation.
; 111          0111   1 !
; 112          0112   1 !        V03-018 LMP0149         L. Mark Pilant,         16-Sep-1983  11:25
; 113          0113   1 !                Make sure the header is turned back to the primary header
; 114          0114   1 !                after a build ACL operation.
;  115      0115   1 !
;  116      0116   1 !     V03-017 LMP0151           L. Mark Pilant,           12-Sep-1983  9:31
;  117      0117   1 !             Force ALARM and AUDIT ACEs to be positioned at the front of
;  118      0118   1 !             the ACL.
;  119      0119   1 !
;  120      0120   1 !     V03-016 LMP0130           L. Mark Pilant,           28-Jul-1983  10:29
;  121      0121   1 !             Correct a bug that caused an accvio if the last header in the
;  122      0122   1 !             chain was completely filled with map pointers, and an ACL was
;  123      0123   1 !             added.
;  124      0124   1 !
;  125      0125   1 !     V03-15  LMP0105           L. Mark Pilant,           24-Apr-1983  22:04
;  126      0126   1 !             Correct a bug in the segment size calculation when saving
;  127      0127   1 !             protected ACEs during a delete ACL operation.
;  128      0128   1 !
;  129      0129   1 !     V03-014 LMP0102           L. Mark Pilant,           18-Apr-1983  16:57
;  130      0130   1 !             Return an error if an attempt is made to modify an entry in
;  131      0131   1 !             a non-existant ACL.
;  132      0132   1 !
;  133      0133   1 !     V03-013 LMP0097           L. Mark Pilant,           30-Mar-1983  16:18
;  134      0134   1 !             Remove reference to unneeded require file.
;  135      0135   1 !
;  136      0136   1 !     V03-012 LMP0088           L. Mark Pilant,           17-Mar-1983  12:28
;  137      0137   1 !             Add support for reading a single ACE.
;  138      0138   1 !
;  139      0139   1 !     V03-011 LMP0079           L. Mark Pilant,           11-Feb-1983  8:56
;  140      0140   1 !             Correct a problem that caused the system to crash if the
;  141      0141   1 !             ACL builder needed to create an extension header.
;  142      0142   1 !
;  143      0143   1 !     V03-010 LMP0073           L. Mark Pilant,           20-Jan-1983  10:07
;  144      0144   1 !             Use the reserved area offset as the upper limit to the
;  145      0145   1 !             size of the ACL segment in the header.  Also, modify the
;  146      0146   1 !             usage of the ACL context longword in the FIB. (0:23 represent
;  147      0147   1 !             the ACE number and 24:31 represents the type of the last
;  148      0148   1 !             ACE found.)
;  149      0149   1 !
;  150      0150   1 !     V03-009 LMP0068           L. Mark Pilant,           21-Dec-1982  11:57
;  151      0151   1 !             Eliminate the code in the ACL building routine that builds
;  152      0152   1 !             the in-core ACL from the file headers as it is now done in
;  153      0153   1 !             the FCB initialization routine FILL_FCB.
;  154      0154   1 !
;  155      0155   1 !     V03-008 LMP0064           L. Mark Pilant,           9-Dec-1982  16:45
;  156      0156   1 !             Change the manner in which errors are signaled by the ACL
;  157      0157   1 !             handling routines.  If an error occurs, the ACE size is
;  158      0158   1 !             set to zero (along with the type code), and the error
;  159      0159   1 !             code is returned in the flag field.
;  160      0160   1 !
;  161      0161   1 !     V03-007 LMP0058           L. Mark Pilant,           9-Nov-1982  13:00
;  162      0162   1 !             Modify the ACL building routine to fix some boundary
;  163      0163   1 !             conditions.  In addition, get the FIB address from the
;  164      0164   1 !             correct place in the dispatcher.
;  165      0165   1 !
;  166      0166   1 !     V03-006 LMP0054           L. Mark Pilant,           25-Oct-1982  17:15
;  167      0167   1 !             Add an attribute to return the length of an ACL.
;  168      0168   1 !
;  169      0169   1 !     V03-005 LMP0048           L. Mark Pilant,           5-Oct-1982  8:55
;  170      0170   1 !             Modify the ACL building routine to correctly build the file
;  171      0171   1 !             header ACL from a null ACL.
;   172          0172   1 !
;   173          0173   1 !         V03-004 LMP0046         L. Mark Pilant,         24-Sep-1982  8:55
;   174          0174   1 !                 Correct various boundary conditions.  Also modify the build
;   175          0175   1 !                 routine to pad out the ACL area with nulls.
;   176          0176   1 !
;   177          0177   1 !         V03-003 LMP0042         L. Mark Pilant,          9-Sep-1982  16:50
;   178          0178   1 !                 Modify the ACL builder to point to the primary header after
;   179          0179   1 !                 building the ACL.  This is to fix a problem that caused the
;   180          0180   1 !                 last extension header to be checksumed instead of the primary
;   181          0181   1 !                 when writing attributes.
;   182          0182   1 !
;   183          0183   1 !         V03-002 LMP0041         L. Mark Pilant           7-Sep-1982  11:55
;   184          0184   1 !                 Correct a problem that caused the system to crash when
;   185          0185   1 !                 building the ACL for a multi-header file on a lookup.
;   186          0186   1 !
;   187          0187   1 !**
;    189          0188   1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    190          0189   1 REQUIRE 'SRC$:FCPDEF.B32';
;    191          1180   1
;    192          1181   1
;    193          1182   1 FORWARD ROUTINE
;    194          1183   1         ACL_DISPATCH     : L_NORM,              ! ACL dispatch routine
;    195          1184   1         ACL_BUILDACL     : L_NORM,              ! Build the ACL structures
;    196          1185   1         ACL_COPYACL      : L_NORM,              ! Propagate ACEs between file versions
;    197          1186   1         BUILD_HANDLER    : NOVALUE;             ! Exception handler for build
;    198          1187   1
;    199          1188   1 EXTERNAL ROUTINE
;    200          1189   1         ACL_INIT_QUEUE,                         ! Initialize ACL queue
;    201          1190   1         ACL_ADDENTRY,                           ! Add an Access Control Entry
;    202          1191   1         ACL_DELENTRY,                           ! Delete an ACE
;    203          1192   1         ACL_MODENTRY,                           ! Modify an ACE
;    204          1193   1         ACL_FINDENTRY,                          ! Locate an ACE
;    205          1194   1         ACL_FINDTYPE,                           ! Locate a specific type ACE
;    206          1195   1         ACL_DELETEACL,                          ! Delete the entire ACL
;    207          1196   1         ACL_READACL,                            ! Read the entire ACL
;    208          1197   1         ACL_ACLLENGTH,                          ! Return the length of the ACL
;    209          1198   1         ACL_READACE,                            ! Read a single ACE
;    210          1199   1         ACL_LOCATEACE,                          ! Find ACE by number
;    211          1200   1         ALLOC_PAGED      : L_NORM,              ! Allocate system dynamic memory
;    212          1201   1         CHECKSUM         : L_NORM,              ! Checksum the modified header
;    213          1202   1         DALLOC_PAGED     : L_NORM,              ! Deallocate system dynamic memory
;    214          1203   1         EXTEND_HEADER    : L_NORM,              ! Extend the present file header
;    215          1204   1         NEXT_HEADER      : L_NORM,              ! Read next extension header
;    216          1205   1         READ_HEADER      : L_NORM,              ! Read header specified by FCB
;    217          1206   1         SEARCH_FCB       : L_NORM,              ! Search the FCB list
;    218          1207   1         SWITCH_VOLUME    : L_NORM,              ! Set the corrent volume
;    219          1208   1         MARK_DIRTY       : L_NORM;              ! Mark buffer for writeback
```

```
1 GLOBAL ROUTINE ACL_DISPATCH (CODE, ADDRESS, COUNT, ACE) : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine is called whenever any ACL processing is to be done.
1 !       The code is checked for validity and if valid, the appropriate routine
1 !       is called to manipulate the ACL's and ACE's.
1 !
1 ! CALLING SEQUENCE:
1 !       ACL_DISPATCH (ARG1, ARG2, ARG3, ARG4)
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: attribute code to determine the action to perform
1 !       ARG2: address of the header ACL area, 0 if none
1 !       ARG3: size of the attribute
1 !       ARG4: address of the user Access Control Entry
1 !
1 ! IMPLICIT INPUTS:
1 !       IO_PACKET: address of the I/O packet for this operation
1 !
1 ! OUTPUT PARAMETERS:
1 !       none
1 !
1 ! IMPLICIT OUTPUTS:
1 !       none
1 !
1 ! ROUTINE VALUE:
1 !       1 if successful
1 !       0 otherwise
1 !
1 ! SIDE EFFECTS:
1 !       Appropriate action routine called, possible header modification may
1 !       result.
1 !
1 !--
1
2 BEGIN
2
2 BIND_COMMON;
2
2 MAP
2       ADDRESS         : REF BBLOCK,           ! Address of the header area
2       ACE             : REF BBLOCK;           ! Address of the user ACE
2
2 LOCAL
2       LOCAL_STATUS;                           ! Status of attribute processing
2
2 ! Do some initial checking to see if the request can be blown away up front.
2
2 LOCAL_STATUS = SS$_NORMAL;
2 CASE .CODE + 1 FROM ATR$C_ADDACLENT TO ATR$C_READACE OF
2 SET
2     [ATR$C_ADDACLENT,
2      ATR$C_DELACLENT,
2      ATR$C_MODACLENT,
2      ATR$C_FNDACLENT,
2      ATR$C_FNDACLTYP]:
3         BEGIN
3         IF .PRIMARY_FCB[FCB$V_BADACL] THEN LOCAL_STATUS = SS$_BADACL;
3         IF .PRIMARY_FCB[FCB$W_FID_NUM] EQL FID$C_INDEXF
3         AND .PRIMARY_FCB[FCB$B_FID_NMX] EQL 0
3         THEN LOCAL_STATUS = SS$_NOACLSUPPORT;
2         END;
2
2     [ATR$C_DELETEACL]:
3         BEGIN
3         IF .PRIMARY_FCB[FCB$W_FID_NUM] EQL FID$C_INDEXF
3         AND .PRIMARY_FCB[FCB$B_FID_NMX] EQL 0
3         THEN LOCAL_STATUS = SS$_NOACLSUPPORT;
2         END;
2
2     [INRANGE]:  0
2 TES;
2
2 ! Dispatch to the appropriate action routine based upon the attribute code.
2
2 IF .LOCAL_STATUS
2 THEN
3     BEGIN
4     LOCAL_STATUS = (
4         CASE .CODE + 1 FROM ATR$C_ADDACLENT TO ATR$C_READACE OF
4         SET
4             [ATR$C_ADDACLENT]:
5                 BEGIN
5                 IF NOT .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
5                 THEN ACL_INIT_QUEUE (PRIMARY_FCB[FCB$R_ORB]);
5                 IF .CURRENT_FIB[FIB$L_ACL_STATUS]
5                 THEN ACL_ADDENTRY (PRIMARY_FCB[FCB$L_ACLFL], CURRENT_FIB[FIB$L_ACLCTX], .COUNT, .ACE)
5                 ELSE 1
4                 END;
4
4             [ATR$C_DELACLENT]:
4                 IF NOT .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
4                 THEN SS$_ACLEMPTY
5                 ELSE (IF .CURRENT_FIB[FIB$L_ACL_STATUS]
5                       THEN ACL_DELENTRY (PRIMARY_FCB[FCB$L_ACLFL], CURRENT_FIB[FIB$L_ACLCTX], .COUNT, .ACE)
4                       ELSE 1);
4
4             [ATR$C_MODACLENT]:
4                 IF NOT .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
4                 THEN SS$_ACLEMPTY
5                 ELSE (IF .CURRENT_FIB[FIB$L_ACL_STATUS]
5                       THEN ACL_MODENTRY (PRIMARY_FCB[FCB$L_ACLFL], CURRENT_FIB[FIB$L_ACLCTX], .COUNT, .ACE)
4                       ELSE 1);
4
4             [ATR$C_FNDACLENT]:
4                 IF NOT .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
4                 THEN SS$_ACLEMPTY
4                 ELSE ACL_FINDENTRY (PRIMARY_FCB[FCB$L_ACLFL], CURRENT_FIB[FIB$L_ACLCTX], .COUNT, .ACE, 0);
4
4             [ATR$C_FNDACLTYP]:
4                 IF NOT .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
```

```
;  335        1323 4                THEN SS$_ACLEMPTY
;  336        1324 4                ELSE ACL_FINDTYPE (PRIMARY_FCB[FCB$L_ACLFL], CURRENT_FIB[FIB$L_ACLCTX], .COUNT, .ACE, 0);
;  337        1325 4
;  338        1326 4            [ATR$C_DELETEACL]:
;  339        1327 4                IF NOT .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
;  340        1328 4                THEN SS$_NORMAL
;  341        1329 5                ELSE (IF .CURRENT_FIB[FIB$L_ACL_STATUS]
;  342        1330 5                        THEN ACL_DE[ETEACL (PRIMARY_FCB[FCB$L_ACLFL], CURRENT_FIB[FIB$L_ACLCTX])
;  343        1331 4                        ELSE 1);
;  344        1332 4
;  345        1333 4            [ATR$C_READACL]:
;  346        1334 4                IF NOT .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
;  347        1335 4                THEN SS$_ACLEMPTY
;  348        1336 4                ELSE ACL_READACL (PRIMARY_FCB[FCB$L_ACLFL], CURRENT_FIB[FIB$L_ACLCTX], .COUNT, .ACE);
;  349        1337 4
;  350        1338 4            [ATR$C_ACLLENGTH]:
;  351        1339 4                IF NOT .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
;  352        1340 4                THEN
;  353        1341 5                    BEGIN
;  354        1342 5                    CH$FILL (0, .COUNT, .ACE);
;  355        1343 5                    SS$_NORMAL
;  356        1344 5                    END
;  357        1345 4                ELSE ACL_ACLLENGTH (PRIMARY_FCB[FCB$L_ACLFL], CURRENT_FIB[FIB$L_ACLCTX], .COUNT, .ACE);
;  358        1346 4
;  359        1347 4            [ATR$C_READACE]:
;  360        1348 4                IF NOT .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
;  361        1349 4                THEN SS$_ACLEMPTY
;  362        1350 4                ELSE ACL_READACE (PRIMARY_FCB[FCB$L_ACLFL], CURRENT_FIB[FIB$L_ACLCTX], .COUNT, .ACE);
;  363        1351 4
;  364        1352 4
;  365        1353 4            [INRANGE]:  SS$_BADPARAM;
;  366        1354 4
;  367        1355 3        TES);
;  368        1356 2    END;
;  369        1357 2
;  370        1358 2 IF .CURRENT_FIB[FIB$L_ACL_STATUS]
;  371        1359 2 THEN CURRENT_FIB[FIB$[_AC[_STATUS] = .LOCAL_STATUS;
;  372        1360 2
;  373        1361 2 RETURN 1;
;  374        1362 2
;  375        1363 1 END;                                            ! End of routine ACL_DISPATCH


                                                                   .TITLE  ACLCNTRL
                                                                   .IDENT  \V04-000\

                                                                   .EXTRN  ACL_INIT_QUEUE, ACL_ADDENTRY
                                                                   .EXTRN  ACL_DELENTRY, ACL_MODENTRY
                                                                   .EXTRN  ACL_FINDENTRY, ACL_FINDTYPE
                                                                   .EXTRN  ACL_DELETEACL, ACL_READACL
                                                                   .EXTRN  ACL_ACLLENGTH, ACL_READACE
                                                                   .EXTRN  ACL_LOCATEACE, ALLOC_PAGED
                                                                   .EXTRN  CHECKSUM, DALLOC_PAGED
                                                                   .EXTRN  EXTEND_HEADER, NEXT_HEADER
                                                                   .EXTRN  READ_HEADER, SEARCH_FCB
                                                                   .EXTRN  SWITCH_VOLUME, MARK_DIRTY
```

```
                                                                                 .PSECT  $CODE$,NOWRT,2

                                                              01FC 00000         .ENTRY  ACL_DISPATCH, Save R2,R3,R4,R5,R6,R7,R8      ; 1209
                                             57        08   AA  9E 00002         MOVAB   8(BASE), R7                                  ; 1247
                                             58        10   AA  9E 00006         MOVAB   16(BASE), R8                                 ;
                                             56             01  D0 0000A         MOVL    #1, LOCAL_STATUS                             ; 1260
                               08            1E        04   AC  CF 0000D         CASEL   CODE, #30, #8                                ; 1261
              0012           0012          0012           0012     00012 1$:     .WORD   2$-1$,-                                      ;
              0032           0032          001F           0012     0001A                 2$-1$,-                                      ;
                                                          0032     00022                 2$-1$,-                                      ;
                                                                                         2$-1$,-                                      ;
                                                                                         2$-1$,-                                      ;
                                                                                         3$-1$,-                                      ;
                                                                                         4$-1$,-                                      ;
                                                                                         4$-1$,-                                      ;
                                                                                         4$-1$                                        ;
                                             50             67  D0 00024 2$:     MOVL    (R7), R0                                     ; 1269
                                                       22   A0  95 00027         TSTB    34(R0)                                       ;
                                                            05  18 0002A         BGEQ    3$                                           ;
                                             56      0E3A   8F  3C 0002C         MOVZWL  #3642, LOCAL_STATUS                          ;
                                             50             67  D0 00031 3$:     MOVL    (R7), R0                                     ; 1277
                                             01        24   A0  B1 00034         CMPW    36(R0), #1                                   ;
                                                            0A  12 00038         BNEQ    4$                                           ;
                                                       29   A0  95 0003A         TSTB    41(R0)                                       ; 1278
                                                            05  12 0003D         BNEQ    4$                                           ;
                                             56      22BC   8F  3C 0003F         MOVZWL  #8892, LOCAL_STATUS                          ; 1279
                                             03             56  E8 00044 4$:     BLBS    LOCAL_STATUS, 5$                             ; 1287
                                                          0163  31 00047         BRW     31$                                          ;
                               08            1E        04   AC  CF 0004A 5$:     CASEL   CODE, #30, #8                                ; 1291
              0089           0064          003F           0012     0004F 6$:     .WORD   7$-6$,-                                      ;
              010B           00EC          00CB           00AA     00057                 9$-6$,-                                      ;
                                                          0137     0005F                 10$-6$,-                                     ;
                                                                                         12$-6$,-                                     ;
                                                                                         14$-6$,-                                     ;
                                                                                         17$-6$,-                                     ;
                                                                                         20$-6$,-                                     ;
                                                                                         23$-6$,-                                     ;
                                                                                         27$-6$                                       ;
                                             50             67  D0 00061 7$:     MOVL    (R7), R0                                     ; 1295
                               08        63  A0             01  E0 00064         BBS     #1, 99(R0), 8$                               ;
                                                       58   A0  9F 00069         PUSHAB  88(R0)                                       ; 1296
                                      0000G  CF             01  FB 0006C         CALLS   #1, ACL_INIT_QUEUE                           ;
                                             50             68  D0 00071 8$:     MOVL    (R8), R0                                     ; 1297
                                             46        34   A0  E9 00074         BLBC    52(R0), 11$                                  ;
                                             7E        0C   AC  7D 00078         MOVQ    COUNT, -(SP)                                 ; 1298
                                                       30   A0  9F 0007C         PUSHAB  48(R0)                                       ;
                               7E            67  00000080   8F  C1 0007F         ADDL3   #128, (R7), -(SP)                            ;
                                      0000G  CF             04  FB 00087         CALLS   #4, ACL_ADDENTRY                             ;
                                                            69  11 0008C         BRB     13$                                          ;
                                             50             67  D0 0008E 9$:     MOVL    (R7), R0                                     ; 1303
                               66        63  A0             01  E1 00091         BBC     #1, 99(R0), 15$                              ;
                                             50             68  D0 00096         MOVL    (R8), R0                                     ; 1305
                                             21        34   A0  E9 00099         BLBC    52(R0), 11$                                  ;
                                             7E        0C   AC  7D 0009D         MOVQ    COUNT, -(SP)                                 ; 1306
                                                       30   A0  9F 000A1         PUSHAB  48(R0)                                       ;
                               7E            67  00000080   8F  C1 000A4         ADDL3   #128, (R7), -(SP)                            ;
                                      0000G  CF             04  FB 000AC         CALLS   #4, ACL_DELENTRY                             ;
```

```
                                              65 11 000B1       BRB      16$                                     ;
                                 50           67 D0 000B3 10$:  MOVL     (R7), R0                                ; 1310
                    41       63  A0           01 E1 000B6       BBC      #1, 99(R0), 15$                         ;
                                 50           68 D0 000BB       MOVL     (R8), R0                                ; 1312
                                 63       34  A0 E9 000BE 11$:  BLBC     52(R0), 18$                             ;
                                 7E       0C  AC 7D 000C2       MOVQ     COUNT, -(SP)                            ; 1313
                                          30  A0 9F 000C6       PUSHAB   48(R0)                                  ;
                    7E           67 00000080  8F C1 000C9       ADDL3    #128, (R7), -(SP)                       ;
                          0000G  CF           04 FB 000D1       CALLS    #4, ACL_MODENTRY                        ;
                                              61 11 000D6       BRB      19$                                     ;
                                 50           67 D0 000D8 12$:  MOVL     (R7), R0                                ; 1317
                    5E       63  A0           01 E1 000DB       BBC      #1, 99(R0), 21$                         ;
                                              7E D4 000E0       CLRL     -(SP)                                   ; 1319
                                 7E       0C  AC 7D 000E2       MOVQ     COUNT, -(SP)                            ;
                    7E           68           30 C1 000E6       ADDL3    #48, (R8), -(SP)                        ;
                    7E           67 00000080  8F C1 000EA       ADDL3    #128, (R7), -(SP)                       ;
                          0000G  CF           05 FB 000F2       CALLS    #5, ACL_FINDENTRY                       ;
                                              5F 11 000F7 13$:  BRB      22$                                     ;
                                 50           67 D0 000F9 14$:  MOVL     (R7), R0                                ; 1322
                    3D       63  A0           01 E1 000FC 15$:  BBC      #1, 99(R0), 21$                         ;
                                              7E D4 00101       CLRL     -(SP)                                   ; 1324
                                 7E       0C  AC 7D 00103       MOVQ     COUNT, -(SP)                            ;
                    7E           68           30 C1 00107       ADDL3    #48, (R8), -(SP)                        ;
                    7E           67 00000080  8F C1 0010B       ADDL3    #128, (R7), -(SP)                       ;
                          0000G  CF           05 FB 00113       CALLS    #5, ACL_FINDTYPE                        ;
                                              6A 11 00118 16$:  BRB      26$                                     ;
                                 50           67 D0 0011A 17$:  MOVL     (R7), R0                                ; 1327
                    48       63  A0           01 E1 0011D       BBC      #1, 99(R0), 24$                         ;
                                 50           68 D0 00122       MOVL     (R8), R0                                ; 1329
                                 41       34  A0 E9 00125 18$:  BLBC     52(R0), 24$                             ;
                                          30  A0 9F 00129       PUSHAB   48(R0)                                  ; 1330
                    7E           67 00000080  8F C1 0012C       ADDL3    #128, (R7), -(SP)                       ;
                          0000G  CF           02 FB 00134       CALLS    #2, ACL_DELETEACL                       ;
                                              6F 11 00139 19$:  BRB      30$                                     ;
                                 50           67 D0 0013B 20$:  MOVL     (R7), R0                                ; 1334
                    4B       63  A0           01 E1 0013E 21$:  BBC      #1, 99(R0), 28$                         ;
                                 7E       0C  AC 7D 00143       MOVQ     COUNT, -(SP)                            ; 1336
                    7E           68           30 C1 00147       ADDL3    #48, (R8), -(SP)                        ;
                    7E           67 00000080  8F C1 0014B       ADDL3    #128, (R7), -(SP)                       ;
                          0000G  CF           04 FB 00153       CALLS    #4, ACL_READACL                         ;
                                              50 11 00158 22$:  BRB      30$                                     ;
                                 50           67 D0 0015A 23$:  MOVL     (R7), R0                                ; 1339
                    0D       63  A0           01 E0 0015D       BBS      #1, 99(R0), 25$                         ;
0C    AC            00           6E           00 2C 00162       MOVC5    #0, (SP), #0, COUNT, @ACE               ; 1342
                                          10  BC    00168                                                        ;
                                 56           01 D0 0016A 24$:  MOVL     #1, LOCAL_STATUS                        ; 1341
                                              3E 11 0016D       BRB      31$                                     ;
                                 7E       0C  AC 7D 0016F 25$:  MOVQ     COUNT, -(SP)                            ; 1345
                    7E           68           30 C1 00173       ADDL3    #48, (R8), -(SP)                        ;
                    7E           67 00000080  8F C1 00177       ADDL3    #128, (R7), -(SP)                       ;
                          0000G  CF           04 FB 0017F       CALLS    #4, ACL_ACLLENGTH                       ;
                                              24 11 00184 26$:  BRB      30$                                     ;
                                 50           67 D0 00186 27$:  MOVL     (R7), R0                                ; 1348
                    07       63  A0           01 E0 00189       BBS      #1, 99(R0), 29$                         ;
                                 56     09D0  8F 3C 0018E 28$:  MOVZWL   #2512, LOCAL_STATUS                     ;
                                              18 11 00193       BRB      31$                                     ;
                                 7E       0C  AC 7D 00195 29$:  MOVQ     COUNT, -(SP)                            ; 1350
```

```
                7E        68           30  C1 00199          ADDL3   #48, (R8), -(SP)                        ;
                7E        67 00000080  8F  C1 0019D          ADDL3   #128, (R7), -(SP)                       ;
                     0000G CF          04  FB 001A5          CALLS   #4, ACL_READACE                         ;
                          56           50  D0 001AA 30$:     MOVL    R0, LOCAL_STATUS                        ;
                          50           68  D0 001AD 31$:     MOVL    (R8), R0                                ; 1358
                          04     34    A0  E9 001B0          BLBC    52(R0), 32$                             ;
                     34   A0           56  D0 001B4          MOVL    LOCAL_STATUS, 52(R0)                    ; 1359
                          50           01  D0 001B8 32$:     MOVL    #1, R0                                  ; 1361
                                           04 001BB          RET                                             ; 1363

; Routine Size:  444 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 GLOBAL ROUTINE ACL_BUILDACL (FIRST_FCB) : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine, builds the file header ACL from the in core ACL.
1 !
1 !       It should be noted that during the time that the headers are being
1 !       written, should the system crash, the ACL in the headers will be
1 !       corrupted.
1 !
1 ! CALLING SEQUENCE:
1 !       ACL_BUILDACL (ARG1)
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: Address of the first FCB segment (this is usually the same as
1 !             PRIMARY_FCB)
1 !
1 ! IMPLICIT INPUTS:
1 !       none
1 !
1 ! OUTPUT PARAMETERS:
1 !       none
1 !
1 ! IMPLICIT OUTPUTS:
1 !       none
1 !
1 ! ROUTINE VALUE:
1 !       1 if successful
1 !       0 otherwise
1 !
1 ! SIDE EFFECTS:
1 !       The file headers are modified to reflect the state of the current
1 !       in core ACL.
1 !
1 !--
1
2 BEGIN
2
2 MAP
2       FIRST_FCB       : REF BBLOCK;           ! Address of the first FCB segment
2
2 BIND_COMMON;
2
2 LOCAL
2       ACL_POINTER     : REF BBLOCK,           ! Pointer to current ACL segment
2       ACE_POINTER     : REF BBLOCK,           ! Pointer to current ACE
2       ACE             : REF BBLOCK,           ! Address of current ACE in header
2       ACL_LENGTH,                             ! Size of header segment ACL
2       FCB             : REF BBLOCK,           ! FCB segment address
2       HEADER          : REF BBLOCK,           ! Address of primary header
2       NEW_HEADER      : REF BBLOCK,           ! Address of extension header
2       FIRST_ACE;                              ! True if first ACE in header
2
2 ! Set up the exception handler in case the header reading routine have trouble.
2
```

```
3 BEGIN
3 BUILTIN FP;
3 .FP = BUILD_HANDLER;
2 END;
2
2 ! Checksum the current file header in case it has been modified.
2
2 IF .FILE_HEADER NEQ 0
2 THEN
2     IF NOT CHECKSUM (.FILE_HEADER)
2     THEN MARK_DIRTY (.FILE_HEADER);
2
2 ! Get the first file header.
2
2 FCB = .FIRST_FCB;
2 IF NOT .BBLOCK[FCB[FCB$R_ORB], ORB$V_ACL_QUEUE] THEN RETURN 1;
2 NEW_HEADER = HEADER = READ_HEADER (0, .FCB);
2
2 ! Build the file header ACL from the in core version. Set up on the
2 ! first header and the ACL listhead. Note that in the first header,
2 ! we guarantee space for at least one maximal length map pointer.
2 ! The contiguous files managed by the file system (directories, etc.)
2 ! depend on the availability of this map pointer.
2
2 IF .FIRST_FCB[FCB$L_ACLFL] NEQA FIRST_FCB[FCB$L_ACLFL]
2 THEN
3     BEGIN
3     ACL_POINTER = .FIRST_FCB[FCB$L_ACLFL];
3     ACE_POINTER = ACL_POINTER[ACL$L_LIST];
3     HEADER[FH2$B_ACOFFSET] = .HEADER[FH2$B_RSOFFSET];
4     ACL_LENGTH = (.HEADER[FH2$B_RSOFFSET] -
4                   .HEADER[FH2$B_MPOFFSET] -
4                   .HEADER[FH2$B_MAP_INUSE]) * 2;
3     ACE = .HEADER + .HEADER[FH2$B_RSOFFSET] * 2 - .ACL_LENGTH;
3     IF .HEADER[FH2$B_MAP_INUSE] LSSU FM2$C_LENGTH3/2
3     THEN
4         BEGIN
4         ACE = CH$FILL (0, FM2$C_LENGTH3 - .HEADER[FH2$B_MAP_INUSE]*2, .ACE);
4         ACL_LENGTH = .ACL_LENGTH - (FM2$C_LENGTH3 - .HEADER[FH2$B_MAP_INUSE]*2);
3         END;
3     FIRST_ACE = 1;
3
3 ! Scan through the headers until we find one with sufficient free space
3 ! to hold the next ACE.
3
3     WHILE 1
3     DO
4         BEGIN
4         UNTIL .ACE_POINTER[ACE$B_SIZE] LEQU .ACL_LENGTH
4         DO
5             BEGIN
5             CH$FILL (0, .ACL_LENGTH, .ACE);
5             HEADER[FH2$V_BADACL] = 1;
5             CHECKSUM (.HEADER);
5             MARK_DIRTY (.HEADER);
5             NEW_HEADER = NEXT_HEADER (.HEADER, .FCB);
5             IF .NEW_HEADER EQL 0
```

```
ACLC
V04-

;

;

; Si
; Ru
; El
; Li
; Le
; Me
; Co
;  491        1478  5                 THEN NEW_HEADER = EXTEND_HEADER (.CURRENT_FIB, .HEADER, .FCB);
;  492        1479  5                 HEADER = .NEW_HEADER;
;  493        1480  5                 FCB = .FCB[FCB$L_EXFCB];
;  494        1481  5                 HEADER[FH2$B_ACOFFSET] = .HEADER[FH2$B_RSOFFSET];
;  495        1482  6                 ACL_LENGTH = (.HEADER[FH2$B_RSOFFSET] =
;  496        1483  6                               .HEADER[FH2$B_MPOFFSET] -
;  497        1484  5                               .HEADER[FH2$B_MAP_INUSE]) * 2;
;  498        1485  5                 ACE = .HEADER + .HEADER[FH2$B_RSOFFSET] * 2 - .ACL_LENGTH;
;  499        1486  5                 FIRST_ACE = 1;                         ! First ACE in header
;  500        1487  4                 END;
;  501        1488  4
;  502        1489  4 ! Having found a header with free space, mark it as belonging to the ACL.
;  503        1490  4
;  504        1491  4             IF .FIRST_ACE
;  505        1492  4             THEN
;  506        1493  5                 BEGIN
;  507        1494  5                 HEADER[FH2$B_ACOFFSET] = .HEADER[FH2$B_RSOFFSET] - .ACL_LENGTH/2;
;  508        1495  5                 FIRST_ACE = 0;
;  509        1496  4                 END;
;  510        1497  4
;  511        1498  4 ! Copy the ACE into the header and advance the pointers.
;  512        1499  4
;  513        1500  4             CH$MOVE (.ACE_POINTER[ACE$B_SIZE], .ACE_POINTER, .ACE);
;  514        1501  4             ACE = .ACE + .ACE_POINTER[ACE$B_SIZE];
;  515        1502  4             ACL_LENGTH = .ACL_LENGTH - .ACE_POINTER[ACE$B_SIZE];
;  516        1503  4             ACE_POINTER = .ACE_POINTER + .ACE_POINTER[ACE$B_SIZE];
;  517        1504  4             IF .ACE_POINTER GEQA .ACL_POINTER + .ACL_POINTER[ACL$W_SIZE]
;  518        1505  4             THEN
;  519        1506  5                 BEGIN
;  520        1507  5                 ACL_POINTER = .ACL_POINTER[ACL$L_FLINK];
;  521        1508  5                 IF .ACL_POINTER EQLA FIRST_FCB[FCB$L_ACLFL] THEN EXITLOOP;
;  522        1509  5                 ACE_POINTER = ACL_POINTER[ACL$L_LIST];
;  523        1510  4                 END;
;  524        1511  3             END;
;  525        1512  3
;  526        1513  3 ! Recover any unused ACL space from this header by sliding the ACL down
;  527        1514  3 ! to the end of the header. Clear the odd byte at the end of the ACL
;  528        1515  3 ! if there is one.
;  529        1516  3
;  530        1517  3     IF TESTBITSC (ACL_LENGTH<0,1>) THEN (.ACE)<0,8> = 0;
;  531        1518  3     CH$MOVE ((.HEADER[FH2$B_RSOFFSET] - .HEADER[FH2$B_ACOFFSET]) * 2 - .ACL_LENGTH,
;  532        1519  3              .HEADER + .HEADER[FH2$B_ACOFFSET] * 2,
;  533        1520  3              .HEADER + .HEADER[FH2$B_ACOFFSET] * 2 + .ACL_LENGTH);
;  534        1521  3     CH$FILL (0, .ACL_LENGTH, .HEADER + .HEADER[FH2$B_ACOFFSET] * 2);
;  535        1522  3     HEADER[FH2$B_ACOFFSET] = .HEADER[FH2$B_ACOFFSET] + .ACL_LENGTH / 2;
;  536        1523  3     HEADER[FH2$V_BADACL] = 1;
;  537        1524  3     CHECKSUM (.HEADER);
;  538        1525  3     MARK_DIRTY (.HEADER);
;  539        1526  3     END
;  540        1527  2 ELSE
;  541        1528  3     BEGIN
;  542        1529  3
;  543        1530  3 ! If there is not ACL associated with the file, and this is the first (or only)
;  544        1531  3 ! file header, clean it out properly.
;  545        1532  3
;  546        1533  4     CH$FILL (0, (.HEADER[FH2$B_RSOFFSET] -
;  547        1534  3                 .HEADER[FH2$B_ACOFFSET]) * 2,
3              .HEADER + .HEADER[FH2$B_ACOFFSET] * 2);
3
3  ! If there are no extension headers, also re-adjust the ACL offset.  If there
3  ! are extension headers, leave the ACL offset alone as modifying it would
3  ! corrupt the map information in the following headers.
3
3      IF .HEADER[FH2$W_EX_FIDNUM] EQL 0 AND .HEADER[FH2$B_EX_FIDNMX] EQL 0
3      THEN HEADER[FH2$B_ACOFFSET] = .HEADER[FH2$B_RSOFFSET];
3      CHECKSUM (.HEADER);
3      MARK_DIRTY (.HEADER);
2      END;
2
2  ! Clean out the ACL from any remaining headers.
2
2  WHILE 1
2  DO
3      BEGIN
3      NEW_HEADER = NEXT_HEADER (.HEADER, .FCB);
3      FCB = .FCB[FCB$L_EXFCB];
3      IF .NEW_HEADER EQL 0 THEN EXITLOOP;
3      HEADER = .NEW_HEADER;
4      CH$FILL (0, (.HEADER[FH2$B_RSOFFSET] -
3                   .HEADER[FH2$B_ACOFFSET]) * 2,
3              .HEADER + .HEADER[FH2$B_ACOFFSET] * 2);
3      HEADER[FH2$B_ACOFFSET] = .HEADER[FH2$B_RSOFFSET];
3      CHECKSUM (.HEADER);
3      MARK_DIRTY (.HEADER);
2      END;
2
2  ! Now that all of the headers have been written, go back and clear the BADACL
2  ! bit.  This is necessary to avoid a corrupted ACL should the system crash
2  ! while the headers are being re-written.
2
2  IF .HEADER[FH2$W_SEG_NUM] NEQ 0
2  THEN NEW_HEADER = READ_HEADER (0, .FIRST_FCB)
2  ELSE NEW_HEADER = .HEADER;
2  DO
3      BEGIN
3      HEADER = .NEW_HEADER;
3      HEADER[FH2$V_BADACL] = 0;
3      CHECKSUM (.HEADER);
3      MARK_DIRTY (.HEADER);
3      NEW_HEADER = NEXT_HEADER (.HEADER, 0);
3      END
2  UNTIL .NEW_HEADER EQL 0;
2
2  ! Get back the primary file header.
2
2  IF .HEADER[FH2$W_SEG_NUM] NEQ 0
2  THEN READ_HEADER (0, .FIRST_FCB);
2
2  RETURN 1;
2
1  END;                                                    ! End of routine ACL_BUILDACL
```

```
                                             0BFC     00000         .ENTRY   ACL_BUILDACL, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 1364
                                                                             R11                                         :
                                5E             0C  C2 00002         SUBL2    #12, SP                                     :
                                6D    0000V    CF  9E 00005         MOVAB    BUILD_HANDLER, (FP)                         ; 1423
                                         04    AA  D5 0000A         TSTL     4(BASE)                                     ; 1428
                                               13  13 0000D         BEQL     1$                                          :
                                         04    AA  DD 0000F         PUSHL    4(BASE)                                     ; 1430
                         0000G  CF             01  FB 00012         CALLS    #1, CHECKSUM                                :
                                08             50  E8 00017         BLBS     R0, 1$                                      :
                                         04    AA  DD 0001A         PUSHL    4(BASE)                                     ; 1431
                         0000G  CF             01  FB 0001D         CALLS    #1, MARK_DIRTY                              :
                                5B       04    AC  D0 00022 1$:     MOVL     FIRST_FCB, FCB                              ; 1435
                  03        63  AB             01  E0 00026         BBS      #1, 99(FCB), 2$                             ; 1436
                                             022D  31 0002B         BRW      18$                                         :
                                               5B  DD 0002E 2$:     PUSHL    FCB                                         ; 1437
                                               7E  D4 00030         CLRL     -(SP)                                       :
                         0000G  CF             02  FB 00032         CALLS    #2, READ_HEADER                             :
                                59             50  D0 00037         MOVL     R0, HEADER                                  :
                            08  AE             59  D0 0003A         MOVL     HEADER, NEW_HEADER                          :
                                50       04    AC  D0 0003E         MOVL     FIRST_FCB, R0                               ; 1445
                                51     0080    C0  9E 00042         MOVAB    128(R0), R1                                 :
                                51     0080    C0  D1 00047         CMPL     128(R0), R1                                 :
                                               03  12 0004C         BNEQ     3$                                          :
                                             0154  31 0004E         BRW      12$                                         :
                                57     0080    C0  D0 00051 3$:     MOVL     128(R0), ACL_POINTER                        ; 1448
                                58       0C    A7  9E 00056         MOVAB    12(R7), ACE_POINTER                         ; 1449
                            02  A9       03    A9  90 0005A         MOVB     3(HEADER), 2(HEADER)                        ; 1450
                                50       03    A9  9A 0005F         MOVZBL   3(HEADER), R0                               ; 1452
                                51       01    A9  9A 00063         MOVZBL   1(HEADER), R1                               :
                                50             51  C2 00067         SUBL2    R1, R0                                      :
                                52       3A    A9  9A 0006A         MOVZBL   58(HEADER), R2                              ; 1453
                                50             52  C2 0006E         SUBL2    R2, R0                                      :
                  56            50             01  78 00071         ASHL     #1, R0, ACL_LENGTH                          :
                                50       03    A9  9A 00075         MOVZBL   3(HEADER), R0                               ; 1454
                                50           6940  3E 00079         MOVAW    (HEADER)[R0], R0                            :
                  6E            50             56  C3 0007D         SUBL3    ACL_LENGTH, R0, ACE                         :
                                50       3A    A9  9A 00081         MOVZBL   58(HEADER), R0                              ; 1455
                                04             50  91 00085         CMPB     R0, #4                                      :
                                               1C  1E 00088         BGEQU    4$                                          :
                                50             50  CE 0008A         MNEGL    R0, R0                                      ; 1458
                                50             02  C4 0008D         MULL2    #2, R0                                      :
                                50             08  C0 00090         ADDL2    #8, R0                                      :
      50          00            6E             00  2C 00093         MOVC5    #0, (SP), #0, R0, @ACE                      :
                                         00    BE     00098                                                              :
                                6E             53  D0 0009A         MOVL     R3, ACE                                     :
                                50       3A    A9  9A 0009D         MOVZBL   58(HEADER), R0                              ; 1459
                                56       F8  A640  3E 000A1         MOVAW    -8(ACL_LENGTH)[R0], ACL_LENGTH              :
                            04  AE             01  D0 000A6 4$:     MOVL     #1, FIRST_ACE                               ; 1461
      56          68            08             00  ED 000AA 5$:     CMPZV    #0, #8, (ACE_POINTER), ACL_LENGTH           ; 1469
                                               69  1B 000AF         BLEQU    7$                                          :
      56          00            6E             00  2C 000B1         MOVC5    #0, (SP), #0, ACL_LENGTH, @ACE              ; 1472
                                         00    BE     000B6                                                              :
                            35  A9             08  88 000B8         BISB2    #8, 53(HEADER)                              ; 1473
                                               59  DD 000BC         PUSHL    HEADER                                      ; 1474
                         0000G  CF             01  FB 000BE         CALLS    #1, CHECKSUM                                :
```

```
                                                59 DD 000C3       PUSHL   HEADER                                        ; 1475
                         0000G   CF             01 FB 000C5       CALLS   #1, MARK_DIRTY                                ;
                                         0A00   8F BB 000CA       PUSHR   #^M<R9,R11>                                   ; 1476
                         0000G   CF             02 FB 000CE       CALLS   #2, NEXT_HEADER                               ;
                            08   AE             50 D0 000D3       MOVL    R0, NEW_HEADER                                ;
                                                10 12 000D7       BNEQ    6$                                            ; 1477
                                         0A00   8F BB 000D9       PUSHR   #^M<R9,R11>                                   ; 1478
                                           10   AA DD 000DD       PUSHL   16(BASE)                                      ;
                         0000G   CF             03 FB 000E0       CALLS   #3, EXTEND_HEADER                             ;
                            08   AE             50 D0 000E5       MOVL    R0, NEW_HEADER                                ;
                                 59        08   AE D0 000E9 6$:   MOVL    NEW_HEADER, HEADER                            ; 1479
                                 5B        0C   AB D0 000ED       MOVL    12(FCB), FCB                                  ; 1480
                            02   A9        03   A9 90 000F1       MOVB    3(HEADER), 2(HEADER)                          ; 1481
                                 50        03   A9 9A 000F6       MOVZBL  3(HEADER), R0                                 ; 1483
                                 51        01   A9 9A 000FA       MOVZBL  1(HEADER), R1                                 ;
                                 50             51 C2 000FE       SUBL2   R1, R0                                        ;
                                 52        3A   A9 9A 00101       MOVZBL  58(HEADER), R2                                ; 1484
                                 50             52 C2 00105       SUBL2   R2, R0                                        ;
                   56            50             01 78 00108       ASHL    #1, R0, ACL_LENGTH                            ;
                                 50        03   A9 9A 0010C       MOVZBL  3(HEADER), R0                                 ; 1485
                                 50      6940      3E 00110       MOVAW   (HEADER)[R0], R0                              ;
                   6E            50             56 C3 00114       SUBL3   ACL_LENGTH, R0, ACE                           ;
                                                8C 11 00118       BRB     4$                                            ; 1486
                   0D                      04   AE E9 0011A 7$:   BLBC    FIRST_ACE, 8$                                 ; 1491
                   50            56             02 C7 0011E       DIVL3   #2, ACL_LENGTH, R0                            ; 1494
            02     A9       03   A9             50 83 00122       SUBB3   R0, 3(HEADER), 2(HEADER)                      ;
                                           04   AE D4 00128       CLRL    FIRST_ACE                                     ; 1495
                                 50             68 9A 0012B 8$:   MOVZBL  (ACE_POINTER), R0                             ; 1500
            00     BE            68             50 28 0012E       MOVC3   R0, (ACE_POINTER), @ACE                       ;
                                 50             68 9A 00133       MOVZBL  (ACE_POINTER), R0                             ; 1501
                                 6E             50 C0 00136       ADDL2   R0, ACE                                       ;
                                 50             68 9A 00139       MOVZBL  (ACE_POINTER), R0                             ; 1502
                                 56             50 C2 0013C       SUBL2   R0, ACL_LENGTH                                ;
                                 50             68 9A 0013F       MOVZBL  (ACE_POINTER), R0                             ; 1503
                                 58             50 C0 00142       ADDL2   R0, ACE_POINTER                               ;
                                 50        08   A7 3C 00145       MOVZWL  8(ACL_POINTER), R0                            ; 1504
                                 50             57 C0 00149       ADDL2   ACL_POINTER, R0                               ;
                                 50             58 D1 0014C       CMPL    ACE_POINTER, R0                               ;
                                                15 1F 0014F       BLSSU   9$                                            ;
                                 57             67 D0 00151       MOVL    (ACL_POINTER), ACL_POINTER                    ; 1507
                   50       04   AC  00000080   8F C1 00154       ADDL3   #128, FIRST_FCB, R0                           ; 1508
                                 50             57 D1 0015D       CMPL    ACL_POINTER, R0                               ;
                                                07 13 00160       BEQL    10$                                           ;
                                 58        0C   A7 9E 00162       MOVAB   12(R7), ACE_POINTER                           ; 1509
                                              FF41 31 00166 9$:   BRW     5$                                            ; 1466
                   03            56             00 E5 00169 10$:  BBCC    #0, ACL_LENGTH, 11$                           ; 1517
                                           00   BE 94 0016D       CLRB    @ACE                                          ;
                                 51        02   A9 9A 00170 11$:  MOVZBL  2(HEADER), R1                                 ; 1518
                                 50        03   A9 9A 00174       MOVZBL  3(HEADER), R0                                 ;
                                 50             51 C2 00178       SUBL2   R1, R0                                        ;
                                 50             02 C4 0017B       MULL2   #2, R0                                        ;
                                 50             56 C2 0017E       SUBL2   ACL_LENGTH, R0                                ;
                                 51      6941      3E 00181       MOVAW   (HEADER)[R1], R1                              ; 1519
                 6641            61             50 28 00185       MOVC3   R0, (R1), (ACL_LENGTH)[R1]                    ; 1520
                                 50        02   A9 9A 0018A       MOVZBL  2(HEADER), R0                                 ; 1521
                                 57      6940      3E 0018E       MOVAW   (HEADER)[R0], R7                              ;
  56               00            6E             00 2C 00192       MOVC5   #0, (SP), #0, ACL_LENGTH, (R7)                ;
```

```
                                                  67     00197                                                    ;
                                     56            02  C6 00198       DIVL2   #2, R6                              ; 1522
                          02   A9                  56  80 0019B       ADDB2   R6, 2(HEADER)                       ;
                          35   A9                  08  88 0019F       BISB2   #8, 53(HEADER)                      ; 1523
                                                   27  11 001A3       BRB     14$                                 ; 1524
                               51      02          A9  9A 001A5 12$:  MOVZBL  2(HEADER), R1                       ; 1534
                               50      03          A9  9A 001A9       MOVZBL  3(HEADER), R0                       ;
                               50                  51  C2 001AD       SUBL2   R1, R0                              ;
                               50                  02  C4 001B0       MULL2   #2, R0                              ;
                               56                6941  3E 001B3       MOVAW   (HEADER)[R1], R6                    ; 1535
        50        00           6E                  00  2C 001B7       MOVC5   #0, (SP), #0, R0, (R6)              ;
                                                   66     001BC                                                   ;
                                       0E          A9  B5 001BD       TSTW    14(HEADER)                          ; 1541
                                                   0A  12 001C0       BNEQ    14$                                 ;
                                       13          A9  95 001C2       TSTB    19(HEADER)                          ;
                                                   05  12 001C5       BNEQ    14$                                 ;
                          02   A9      03          A9  90 001C7 13$:  MOVB    3(HEADER), 2(HEADER)                ; 1542
                                                   59  DD 001CC 14$:  PUSHL   HEADER                              ; 1543
                       0000G   CF                  01  FB 001CE       CALLS   #1, CHECKSUM                        ;
                                                   59  DD 001D3       PUSHL   HEADER                              ; 1544
                       0000G   CF                  01  FB 001D5       CALLS   #1, MARK_DIRTY                      ;
                                     0A00          8F  BB 001DA       PUSHR   #^M<R9,R11>                         ; 1552
                       0000G   CF                  02  FB 001DE       CALLS   #2, NEXT_HEADER                     ;
                          08   AE                  50  D0 001E3       MOVL    R0, NEW_HEADER                      ;
                               5B      0C          AB  D0 001E7       MOVL    12(FCB), FCB                        ; 1553
                                       08          AE  D5 001EB       TSTL    NEW_HEADER                          ; 1554
                                                   1E  13 001EE       BEQL    15$                                 ;
                               59      08          AE  D0 001F0       MOVL    NEW_HEADER, HEADER                  ; 1555
                               51      02          A9  9A 001F4       MOVZBL  2(HEADER), R1                       ; 1557
                               50      03          A9  9A 001F8       MOVZBL  3(HEADER), R0                       ;
                               50                  51  C2 001FC       SUBL2   R1, R0                              ;
                               50                  02  C4 001FF       MULL2   #2, R0                              ;
                               56                6941  3E 00202       MOVAW   (HEADER)[R1], R6                    ; 1558
        50        00           6E                  00  2C 00206       MOVC5   #0, (SP), #0, R0, (R6)              ;
                                                   66     0020B                                                   ;
                                                   B9  11 0020C       BRB     13$                                 ; 1559
                                       04          A9  B5 0020E 15$:  TSTW    4(HEADER)                           ; 1568
                                                   10  13 00211       BEQL    16$                                 ;
                                       04          AC  DD 00213       PUSHL   FIRST_FCB                           ; 1569
                                                   7E  D4 00216       CLRL    -(SP)                               ;
                       0000G   CF                  02  FB 00218       CALLS   #2, READ_HEADER                     ;
                          08   AE                  50  D0 0021D       MOVL    R0, NEW_HEADER                      ;
                                                   04  11 00221       BRB     17$                                 ;
                          08   AE                  59  D0 00223 16$:  MOVL    HEADER, NEW_HEADER                  ; 1570
                               59      08          AE  D0 00227 17$:  MOVL    NEW_HEADER, HEADER                  ; 1573
                          35   A9                  08  8A 0022B       BICB2   #8, 53(HEADER)                      ; 1574
                                                   59  DD 0022F       PUSHL   HEADER                              ; 1575
                       0000G   CF                  01  FB 00231       CALLS   #1, CHECKSUM                        ;
                                                   59  DD 00236       PUSHL   HEADER                              ; 1576
                       0000G   CF                  01  FB 00238       CALLS   #1, MARK_DIRTY                      ;
                                                   7E  D4 0023D       CLRL    -(SP)                               ; 1577
                                                   59  DD 0023F       PUSHL   HEADER                              ;
                       0000G   CF                  02  FB 00241       CALLS   #2, NEXT_HEADER                     ;
                          08   AE                  50  D0 00246       MOVL    R0, NEW_HEADER                      ;
                                                   DB  12 0024A       BNEQ    17$                                 ; 1579
                                       04          A9  B5 0024C       TSTW    4(HEADER)                           ; 1583
                                                   0A  13 0024F       BEQL    18$                                 ;
```

```
                              04    AC  DD 00251          PUSHL   FIRST_FCB                              ; 1584
                                    7E  D4 00254          CLRL    -(SP)                                  ;
                   0000G  CF        02  FB 00256          CALLS   #2, READ_HEADER                        ;
                          50        01  D0 0025B 18$:     MOVL    #1, R0                                 ; 1586
                                        04 0025E          RET                                            ; 1588

; Routine Size:  607 bytes,    Routine Base:  $CODE$ + 01BC
```

```
1 GLOBAL ROUTINE ACL_COPYACL (OLD_FILE_FCB, NEW_FILE_FCB, OPTION) : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine propagates the ACEs from one file to another.  This
1 !       is done in one of several ways: propagate everything, propagate
1 !       only directory default ACEs, or propagate everything except ACEs
1 !       marked as NOPROPAGATE.
1 !
1 ! CALLING SEQUENCE:
1 !       ACL_APPLYDEF (ARG1, ARG2, ARG3);
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: address of the source file FCB
1 !       ARG2: address of the created file FCB
1 !       ARG3: 0 = copy everything
1 !             1 = only default ACEs
1 !             2 = all except NOPROPAGATE ACEs
1 !
1 ! IMPLICIT INPUTS:
1 !       none
1 !
1 ! OUTPUT PARAMETERS:
1 !       none
1 !
1 ! IMPLICIT OUTPUTS:
1 !       none
1 !
1 ! ROUTINE VALUE:
1 !       1 if successful
1 !       0 otherwise
1 !
1 ! SIDE EFFECTS:
1 !       An ACL for the created file is built using any applicable ACE's
1 !       from the source file.  The file header for the created file is
1 !       then updated to refect this default ACL.
1 !
1 !--
1
2 BEGIN
2
2 MAP
2       OLD_FILE_FCB    : REF BBLOCK,           ! Address of the directory FCB
2       NEW_FILE_FCB    : REF BBLOCK;           ! Address of created file FCB
2
2 BIND_COMMON;
2
2 LOCAL
2       ACL_POINTER     : REF BBLOCK,           ! Directory ACL segment address
2       ACL_LENGTH,                             ! Length of the ACL
2       ACL_AREA        : BBLOCK [MAX_ACL_SIZE], ! buffer to build ACL segment
2       NEW_ACL         : REF BBLOCK,           ! Address of file ACL segment
2       ACE_POINTER     : REF BBLOCK;           ! Address of current ACE
2
2 ! Clean out any existing ACL on the target file.
```

```
;  660      1646  2
;  661      1647  2  ACL_DELETEACL (NEW_FILE_FCB[FCB$L_ACLFL], 0);
;  662      1648  2
;  663      1649  2  ! See if there is any ACL on the source file that must be propagated.
;  664      1650  2
;  665      1651  2  IF .OLD_FILE_FCB[FCB$L_ACLFL] EQLA OLD_FILE_FCB[FCB$L_ACLFL] THEN RETURN 1;
;  666      1652  2
;  667      1653  2  ! See if this is a simple copy operation.
;  668      1654  2
;  669      1655  2  IF .OPTION EQL 0
;  670      1656  2  THEN
;  671      1657  3      BEGIN
;  672      1658  3      ACL_POINTER = .OLD_FILE_FCB[FCB$L_ACLFL];
;  673      1659  3      UNTIL .ACL_POINTER EQLA OLD_FILE_FCB[FCB$L_ACLFL]
;  674      1660  3      DO
;  675      1661  4          BEGIN
;  676      1662  4          NEW_ACL = ALLOC_PAGED (.ACL_POINTER[ACL$W_SIZE], ACL_TYPE);
;  677      1663  4          CH$MOVE (.ACL_POINTER[ACL$W_SIZE], .ACL_POINTER, .NEW_ACL);
;  678      1664  4          INSQUE (.NEW_ACL, .NEW_FILE_FCB[FCB$L_ACLBL]);
;  679      1665  4          ACL_POINTER = .ACL_POINTER[ACL$L_FLINK];
;  680      1666  3          END;
;  681      1667  3      END
;  682      1668  2  ELSE
;  683      1669  3      BEGIN
;  684      1670  3
;  685      1671  3  ! Scan through the source file's ACL and propagate the desired ACE types.
;  686      1672  3
;  687      1673  3      ACL_POINTER = OLD_FILE_FCB[FCB$L_ACLFL];
;  688      1674  3      ACL_LENGTH = 0;
;  689      1675  3      DO
;  690      1676  4          BEGIN
;  691      1677  4          ACL_POINTER = .ACL_POINTER[ACL$L_FLINK];
;  692      1678  4          ACE_POINTER = ACL_POINTER[ACL$L_LIST];
;  693      1679  4          UNTIL .ACE_POINTER GEQA .ACL_POINTER + .ACL_POINTER[ACL$W_SIZE]
;  694      1680  4          DO
;  695      1681  5              BEGIN
;  696      1682  6              IF (.ACE_POINTER[ACE$V_DEFAULT] AND .OPTION EQL 1)
;  697      1683  6              OR (NOT .ACE_POINTER[ACE$V_NOPROPAGATE] AND .OPTION EQL 2)
;  698      1684  5              THEN
;  699      1685  6                  BEGIN
;  700      1686  6                  IF .ACL_LENGTH + .ACE_POINTER[ACE$B_SIZE] GTR MAX_ACL_SIZE
;  701      1687  6                  THEN
;  702      1688  7                      BEGIN
;  703      1689  7                      NEW_ACL = ALLOC_PAGED (ACL$C_LENGTH + .ACL_LENGTH, ACL_TYPE);
;  704      1690  7                      CH$MOVE (.ACL_LENGTH, ACL_AREA, NEW_ACL[ACL$L_LIST]);
;  705      1691  7                      NEW_ACL[ACL$W_SIZE] = ACL$C_LENGTH + .ACL_LENGTH;
;  706      1692  7                      INSQUE (.NEW_ACL, .NEW_FILE_FCB[FCB$L_ACLBL]);
;  707      1693  7                      ACL_LENGTH = 0;
;  708      1694  6                      END;
;  709      1695  6
;  710      1696  6                  CH$MOVE (.ACE_POINTER[ACE$B_SIZE], .ACE_POINTER,
;  711      1697  6                           VECTOR [ACL_AREA, .ACL_LENGTH;, BYTE]);
;  712      1698  6                  IF .OPTION EQL 1
;  713      1699  6                  THEN BBLOCK [VECTOR [ACL_AREA, .ACL_LENGTH;, BYTE], ACE$V_DEFAULT] = 0;
;  714      1700  6                  ACL_LENGTH = .ACL_LENGTH + .ACE_POINTER[ACE$B_SIZE];
;  715      1701  5                  END;
;  716      1702  5              ACE_POINTER = .ACE_POINTER + .ACE_POINTER[ACE$B_SIZE];
;  717          1703  4                END;
;  718          1704  4            END
;  719          1705  3        UNTIL .ACL_POINTER[ACL$L_FLINK] EQLA OLD_FILE_FCB[FCB$L_ACLFL];
;  720          1706  3
;  721          1707  3        IF .ACL_LENGTH GTR 0
;  722          1708  3        THEN
;  723          1709  4            BEGIN
;  724          1710  4            NEW_ACL = ALLOC_PAGED (ACL$C_LENGTH + .ACL_LENGTH, ACL_TYPE);
;  725          1711  4            CH$MOVE (.ACL_LENGTH, ACL_AREA, NEW_ACL[ACE$L_LIST]);
;  726          1712  4            NEW_ACL[ACL$W_SIZE] = ACL$C_LENGTH + .ACL_LENGTH;
;  727          1713  4            INSQUE (.NEW_ACL, .NEW_FILE_FCB[FCB$L_ACLBL]);
;  728          1714  3            END;
;  729          1715  2        END;
;  730          1716  2
;  731          1717  2 1
;  732          1718  1 END;                                            ! End of routine ACL_COPYACL



                                  0BFC 00000              .ENTRY   ACL_COPYACL, Save R2,R3,R4,R5,R6,R7,R8,R9,- ; 1589
                                                                   R11                                         ;
                 5B    0000G  CF    9E 00002              MOVAB    ALLOC_PAGED, R11                            ;
                 5E     FE00  CE    9E 00007              MOVAB    -512(SP), SP                                ;
                              7E    D4 0000C              CLRL     -(SP)                                       ; 1647
   7E      08    AC 00000080  8F    C1 0000E              ADDL3    #128, NEW_FILE_FCB, -(SP)                   ;
         0000G   CF           02    FB 00017              CALLS    #2, ACL_DELETEACL                           ;
                 50       04  AC    D0 0001C              MOVL     OLD_FILE_FCB, R0                            ; 1651
                 50     0080  C0    9E 00020              MOVAB    128(R0), R0                                 ;
                 50           60    D1 00025              CMPL     (R0), R0                                    ;
                              14    13 00028              BEQL     2$                                          ;
                          0C  AC    D5 0002A              TSTL     OPTION                                      ; 1655
                              33    12 0002D              BNEQ     4$                                          ;
                 57           60    D0 0002F              MOVL     (R0), ACL_POINTER                           ; 1658
   50      04    AC 00000080  8F    C1 00032 1$:          ADDL3    #128, OLD_FILE_FCB, R0                      ; 1659
                 50           57    D1 0003B              CMPL     ACL_POINTER, R0                             ;
                              03    12 0003E 2$:          BNEQ     3$                                          ;
                            00CF    31 00040              BRW      14$                                         ;
                              07    DD 00043 3$:          PUSHL    #7                                          ; 1662
                 7E       08  A7    3C 00045              MOVZWL   8(ACL_POINTER), -(SP)                       ;
                 6B           02    FB 00049              CALLS    #2, ALLOC_PAGED                             ;
                 59           50    D0 0004C              MOVL     R0, NEW_ACL                                 ;
   69            67       08  A7    28 0004F              MOVC3    8(ACL_POINTER), (ACL_POINTER), (NEW_ACL)    ; 1663
                 50       08  AC    D0 00054              MOVL     NEW_FILE_FCB, R0                            ; 1664
          0084   D0           69    0E 00058              INSQUE   (NEW_ACL), @132(R0)                         ;
                 57           67    D0 0005D              MOVL     (ACL_POINTER), ACL_POINTER                  ; 1665
                              D0    11 00060              BRB      1$                                          ; 1659
                 57           50    D0 00062 4$:          MOVL     R0, ACL_POINTER                             ; 1673
                              56    D4 00065              CLRL     ACL_LENGTH                                  ; 1674
                 57           67    D0 00067 5$:          MOVL     (ACL_POINTER), ACL_POINTER                  ; 1677
                 58       0C  A7    9E 0006A              MOVAB    12(R7), ACE_POINTER                         ; 1678
                 50       08  A7    3C 0006E 6$:          MOVZWL   8(ACL_POINTER), R0                          ; 1679
                 50           57    C0 00072              ADDL2    ACL_POINTER, R0                             ;
                 50           58    D1 00075              CMPL     ACE_POINTER, R0                             ;
                              65    1E 00078              BGEQU    12$                                         ;
                 06       03  A8    E9 0007A              BLBC     3(ACE_POINTER), 7$                          ; 1682
```

```
                        01       0C    AC   D1 0007E              CMPL     OPTION, #1                                  ;
                                       0B   13 00082              BEQL     8$                                          ;
            4E       03    A8          03   E0 00084 7$:          BBS      #3, 3(ACE_POINTER), 11$                     ; 1683
                        02       0C    AC   D1 00089              CMPL     OPTION, #2                                  ;
                                       48   12 0008D              BNEQ     11$                                         ;
                        50             68   9A 0008F 8$:          MOVZBL   (ACE_POINTER), R0                           ; 1686
                        50             56   C0 00092              ADDL2    ACL_LENGTH, R0                              ;
                 00000200 8F           50   D1 00095              CMPL     R0, #512                                    ;
                                       20   15 0009C              BLEQ     9$                                          ;
                                       07   DD 0009E              PUSHL    #7                                          ; 1689
                                 0C    A6   9F 000A0              PUSHAB   12(ACL_LENGTH)                              ;
                        6B             02   FB 000A3              CALLS    #2, ALLOC_PAGED                             ;
                        59             50   D0 000A6              MOVL     R0, NEW_ACL                                 ;
       0C    A9         6E             56   28 000A9              MOVC3    ACL_LENGTH, ACL_AREA, 12(NEW_ACL)           ; 1690
       08    A9         56             0C   A1 000AE              ADDW3    #12, ACL_LENGTH, 8(NEW_ACL)                 ; 1691
                        50       08    AC   D0 000B3              MOVL     NEW_FILE_FCB, R0                            ; 1692
                     0084 D0           69   0E 000B7              INSQUE   (NEW_ACL), @132(R0)                         ;
                                       56   D4 000BC              CLRL     ACL_LENGTH                                  ; 1693
                        50             68   9A 000BE 9$:          MOVZBL   (ACE_POINTER), R0                           ; 1696
          6E46          68             50   28 000C1              MOVC3    R0, (ACE_POINTER), ACL_AREA[ACL_LENGTH]     ; 1697
                        01       0C    AC   D1 000C6              CMPL     OPTION, #1                                  ; 1698
                                       05   12 000CA              BNEQ     10$                                         ;
                  03 AE46              01   8A 000CC              BICB2    #1, ACL_AREA+3[ACL_LENGTH]                  ; 1699
                        50             68   9A 000D1 10$:         MOVZBL   (ACE_POINTER), R0                           ; 1700
                        56             50   C0 000D4              ADDL2    R0, ACL_LENGTH                              ;
                        50             68   9A 000D7 11$:         MOVZBL   (ACE_POINTER), R0                           ; 1702
                        58             50   C0 000DA              ADDL2    R0, ACE_POINTER                             ;
                                       8F   11 000DD              BRB      6$                                          ; 1679
            50       04    AC 00000080 8F   C1 000DF 12$:         ADDL3    #128, OLD_FILE_FCB, R0                      ; 1705
                        50             67   D1 000E8              CMPL     (ACL_POINTER), R0                           ;
                                       03   13 000EB              BEQL     13$                                         ;
                                     FF77   31 000ED              BRW      5$                                          ;
                                       56   D5 000F0 13$:         TSTL     ACL_LENGTH                                  ; 1707
                                       1E   15 000F2              BLEQ     14$                                         ;
                                       07   DD 000F4              PUSHL    #7                                          ; 1710
                                 0C    A6   9F 000F6              PUSHAB   12(ACL_LENGTH)                              ;
                        6B             02   FB 000F9              CALLS    #2, ALLOC_PAGED                             ;
                        59             50   D0 000FC              MOVL     R0, NEW_ACL                                 ;
       0C    A9         6E             56   28 000FF              MOVC3    ACL_LENGTH, ACL_AREA, 12(NEW_ACL)           ; 1711
       08    A9         56             0C   A1 00104              ADDW3    #12, ACL_LENGTH, 8(NEW_ACL)                 ; 1712
                        50       08    AC   D0 00109              MOVL     NEW_FILE_FCB, R0                            ; 1713
                     0084 D0           69   0E 0010D              INSQUE   (NEW_ACL), @132(R0)                         ;
                        50             01   D0 00112 14$:         MOVL     #1, R0                                      ; 1718
                                            04 00115              RET                                                  ;

; Routine Size:  278 bytes,    Routine Base:  $CODE$ + 041B
```

```
1 ROUTINE BUILD_HANDLER (SIGNAL, MECHANISM) : NOVALUE =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine is the main level condition handler. It stores the
1 !       condition value (FCP error code) in the user status block, unwinds
1 !       and returns from the function that was executing.
1 !
1 ! CALLING SEQUENCE:
1 !       BUILD_HANDLER (ARG1, ARG2)
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: address of signal array
1 !       ARG2: address of mechanism array
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       BUILD_STATUS: receives signal code
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       The stack is unwound to the caller of the ACL builder with the
1 !       error status posted in R0.
1 !
1 !--
1
2 BEGIN
2
2 MAP
2       SIGNAL          : REF BBLOCK,   ! signal array arg
2       MECHANISM       : REF BBLOCK;   ! mechanism array arg
2
2 EXTERNAL ROUTINE
2       SYS$UNWIND      : ADDRESSING_MODE (ABSOLUTE);
2
2 ! Check the signal code. The only permissible ones are SS$_UNWIND, which
2 ! is ignored, and SS$_CMODUSER. The error status is the 16 bit CHMU code.
2 ! If the error value is non-zero, store it in the saved R0 (zero means
2 ! just exit).  Unwind to the caller of the ACL builder and return, causing
2 ! the invoked function to return with failure to the dispatcher.
2 !
2
2 IF .SIGNAL[CHF$L_SIG_NAME] EQL SS$_UNWIND THEN RETURN;
2 IF .SIGNAL[CHF$L_SIG_NAME] NEQ SS$_CMODUSER
2 THEN BUG_CHECK (UNXSIGNAL, FATAL, 'Unexpected signal name in ACP');
2
2 IF .SIGNAL[CHF$L_SIG_ARG1] NEQ 0
2 THEN MECHANISM[CHF$L_MCH_SAVR0] = .SIGNAL[CHF$L_SIG_ARG1];
```

```
;  791          1776  2
;  792          1777  2 SYS$UNWIND (0, 0);
;  793          1778  2
;  794          1779  2 RETURN;
;  795          1780  2
;  796          1781  1 END;                                    ! end of routine BUILD_HANDLER


                                                                          .EXTRN  SYS$UNWIND, BUG$_UNXSIGNAL

                                                       0000 00000 BUILD_HANDLER:
                                                                          .WORD   Save nothing                                 ; 1719
                                       50       04   AC  D0 00002         MOVL    SIGNAL, R0                                   ; 1770
                            00000920   8F       04   A0  D1 00006         CMPL    4(R0), #2336                                 ;
                                                     29  13 0000E         BEQL    3$                                           ;
                            00000424   8F       04   A0  D1 00010         CMPL    4(R0), #1060                                 ; 1771
                                                     04  13 00018         BEQL    1$                                           ;
                                                       FEFF 0001A         BUGW                                                 ; 1772
                                                      0000* 0001C         .WORD   <BUG$_UNXSIGNAL!4>                           ;
                                       51       04   AC  D0 0001E 1$:     MOVL    SIGNAL[, R1                                  ; 1774
                                                08   A1  D5 00022         TSTL    8(R1)                                        ;
                                                     09  13 00025         BEQL    2$                                           ;
                                       50       08   AC  D0 00027         MOVL    MECHANISM, R0                                ; 1775
                                  0C   A0       08   A1  D0 0002B         MOVL    8(R1), 12(R0)                                ;
                                                     7E  7C 00030 2$:     CLRQ    -(SP)                                        ; 1777
                            00000000G  9F            02  FB 00032         CALLS   #2, @#SYS$UNWIND                             ;
                                                         04 00039 3$:     RET                                                  ; 1781

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 0531


;  797          1782  1
;  798          1783  1 END
;  799          1784  0 ELUDOM





;                                      PSECT SUMMARY
;
;       Name                     Bytes                        Attributes
;
;  $CODE$                           1387  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                               Library Statistics
;
;                                            -------- Symbols --------     Pages       Processing
;       File                                 Total    Loaded    Percent     Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1             18619        63          0     1000          00:02.0
;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:ACLCNTRL/OBJ=OBJ$:ACLCNTRL MSRC$:ACLCNTRL/UPDATE=(ENH$:ACLCNTRL)

; Size:          1387 code + 0 data bytes
; Run Time:         00:51.8
; Elapsed Time:     01:46.1
; Lines/CPU Min:      2066
; Lexemes/CPU-Min: 46714
; Memory Used:  356 pages
; Compilation Complete
```

```
; Ro
```

0167 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY